COOL (Computer Outreach Opportunities for Learning) Project

James S. Collofello, Joseph E. Urban,
Mary R. Anderson-Rowland, Faye Navabi, Doris Roman

Arizona State University

Abstract

Although most secondary schools provide some education in computer programming and applications such as spreadsheets and word processors, they are usually deficient in preparing students for careers in software development. The lack of focus on software development topics and project level experiences fails to dispel the "hacker" mentality and "geek-image" myths most secondary school students associate with computing fields. Existing programs provide little insight into professional software development careers such as system analyst, software architect, or system tester. The COOL (Computer Outreach Opportunities for Learning) project is developing and refining an innovative secondary school software development curriculum modules pilot program. This program for secondary teachers is to provide students with a better understanding of the software development field, to dispel misconceptions, and to increase the number and diversity of students continuing their studies in preparation for software development careers, both through recruitment and retention programs. The curriculum models will be integrated into secondary school information technology courses with a focus on targeting K-12 schools with underrepresented minority students.

At the same time, students, including many underrepresented minorities, come to the College of Engineering and Applied Sciences (CEAS) unprepared with the computer skills expected of freshmen. Often the student realizes too late that they were lacking in computer skills and by the time they catch up, the semester is over and the course has to be repeated. These students need a means (placement exam) for determining if they are prepared for the beginning classes requiring computer skills. Based on the placement exam, some students need a basic computer class to get them up to speed to commence with their CEAS major. Therefore, the COOL Project has a freshmen component for incoming ASU students, consisting of a placement test on computer skills expected for the Introduction to Engineering Design course and the beginning course for Java, an Academic Excellence basic computer skills preparation class, Academic Success Workshops (ASW) for the Java course, and a review of the Java course with a view to improve the delivery and student participation in the course. ASW provides concept building and practice exams to assist students with these gatekeeper courses. The placement exam was piloted during the 2002 Minority Summer Bridge Program and Fall 2002 semester Introduction to Java course. Students who did not score well enough on this exam were encouraged to enroll in a special two hour Academic Excellence class on computer science basics. This class was successfully piloted as a one-hour course during the Fall 2001 semester.

Keywords: Computer Basics, Computer Science, Computer Science Placement Exam

I. Introduction

The motivation for this project stems from the need to begin generating excitement and preparing students at earlier ages for information technology careers. Although most secondary schools provide some education in programming languages and many provide advanced placement courses in computer science, they are deficient in providing students with an understanding of what a software development professional does. The lack of focus on professional software development topics, i.e., "software engineering" fails to dispel the "hacker" mentality and "geek-image" myths most secondary school students associate with the field. Female students, in general, are not interested in devoting their entire interest and time to computers and minority students often feel left out of computer groups composed of primarily Caucasians.[1] This situation has contributed to the lack of diversity in the student population taking computer science courses. Recent statistics from the Computer Science Advanced Placement examination administered to secondary school students show a ratio of 10 males to 1 female attempting the full exam and that 84% of the students attempting the full exam are either White or Asian. In addition, a study conducted by the American Association of University Women published in 2000 shows that girls are rejecting computers, computer culture and a variety of other technologies to such an extent that there are fewer girls in computer science now than in the 1980s.[2] Existing secondary school programs provide little insight into professional software development careers such as system analyst, software architect or system tester, and dwell mostly upon simplistic programming tasks. This project will address all of these issues. Although our primary objective is to increase the software development knowledge of all students and to motivate them to pursue further education in the information technology area, our approach should help attract more female and minority students and thus, help improve the diversity of the student population preparing for computer science careers.

The primary driving forces behind the current enhancement strategies are the ABET (Accreditation Board for Engineering and Technology) and CSAC (Computer Science Accreditation Commission) accreditation criteria. These criteria have led the Computer Science and Engineering (CSE) Department to develop outcomes and assessments for all the courses in its degree programs. Based on pre and post course surveys and course evaluation assessments, these courses are continuously improved on a per semester basis by the faculty member and on a per academic year basis by functional area faculty members. During the Fall 2000 semester, Urban offered a one-unit pilot course on Computer Basics to minority engineering program students, which was the basis for an Academic Excellence course on Computer Science Basics [3]. This course covered application skills that would be needed by engineering freshmen, as well as algorithm development. Course performance information was used to determine that the students had the most difficulty with developing objects/classes and algorithm correctness. These topics then formed the core of the Academic Excellence course on Computer Basics. Nineteen of the 21 students completed a course evaluation. The mean ranking on the course was 4.35 and the mean ranking of the instructor was 4.47. The highest ranking of 4.89 was “The instructor exhibited enthusiasm for and interest in the subject.” Given the nature and purpose of this course, this is a very meaningful measure of the success of the course, [3] especially since it

was taught late on Friday afternoon! As an example of the value of this course, consider an American Indian CSE major, with little computer background, who reported that she did well in CSE 110 in the spring due to the preparation she received in this basics course in the fall.

In addition, the criteria used to select the courses CSE 100 Principles of Programming with C++ and CSE 110 Principles of Programming with Java was the high number of students who withdraw from these courses each semester and need to retake the course. In addition, high numbers of students take the courses since it is a prerequisite (for all electrical, computer science, and computer systems engineering students). Since CSE 100 and CSE 110 are the same method of course, one with C^{++} and the other with Java, by concentrating on changes in CSE 110, the course improvements would be immediately applicable to CSE 100, thus leveraging the efforts

II. Background

The CEAS has over 6,600 students enrolled in 12 disciplines. The CEAS has a Minority Engineering Program (MEP) that serves over 750 undergraduate minority CEAS students, and over 800 students participating in several K-12 programs, including ASU MESA, which is in 36 schools.[4,5] The Women in applied Sciences and Engineering (WISE) Program serves nearly 950 undergraduate women.[6,7] The CSE department has two gatekeeper classes, CSE 100 or CSE 110 that many engineering students must pass in order to pursue an engineering degree. During the last three years CSE 100 has averaged 467 students per year and CSE 110 has averaged 353 students. These gatekeeper courses are pivotal to success in electrical engineering, computer science, and computer systems engineering. CSE 100 is required of all electrical engineering majors. Either course, CSE 100 or CSE 110, can be used as the prerequisite courses for the first two courses that are in the CS/E degree program: CSE 200 Concepts of Computer Science and CSE 225 / 226 Assembly Language Programming and Microprocessors. Notice that CSE 200 is a feeder to the software side of the degree programs and CSE 225 / 226 is the feeder to the hardware side of the degree programs.

When secondary schools first became concerned about computer science education many years ago, their primary focus was the Basic programming language and word processing courses. As the computer science field matured and accredited computer science programs began appearing in the universities, more attention was focused on secondary school programs. With few exceptions, most secondary school programs fail to address the issues motivating this project, namely: a) providing students with a broad understanding of software development early in their education; b) dispelling myths about software development careers; and c) motivating students (especially females and minorities) to further study in this field. What is needed in most secondary school curricula are new modules providing students with the necessary breadth to understand software development career opportunities. In 2001, a pilot project, an innovative secondary school software development curriculum modules pilot program, was successful and ready for expansion.[8] The initial content created will form the basis of new modules.

One of the first organizations to address secondary school curriculum issues was the Association for Computing Machinery (ACM). The ACM developed a complete set of secondary school

computer science curriculum guidelines that are still in effect.[2] The guidelines were motivated by the recognition that the need for computer science education in comprehending the information age is analogous to the need for natural science education in understanding the natural world. The ACM curriculum recommendations take the form of a one-year course targeted for students to take in the tenth grade. The curriculum identifies the following seven areas that are appropriate for secondary school students:

1. algorithms
2. programming languages
3. operating systems and user support
4. computer architecture
5. social, ethical and professional context
6. computer applications
7. additional topics (such as software engineering, artificial intelligence and graphics)

More recently, secondary school curriculum guidelines have been developed in Canada and China. In Shanghai schools, computer education is integrated into the core curriculum during the eighth and tenth grades. [9] The curriculum of computer education includes five basic categories:

1. computer basics
2. computer operation and application
3. introduction to commonly used software
4. programming
5. computer impact on modern society and on human society.

In Toronto a new computer and information system curriculum has been developed for the eleventh and twelfth grades.[10] In the eleventh grade, the emphasis is on:

1. problem solving
2. the software development process
3. programming control and data structures
4. documentation and how components of a computer system support software development
5. awareness of computer science careers

In the twelfth grade, the emphasis is on:

1. application of programming and software engineering principles through the design of a real-world programming project
2. continued awareness of both educational and career opportunities in computer science
3. preparation for advanced placement exam

All of the previously described curriculum efforts attempt to provide secondary school students basic programming skills with the more fortunate schools offering the equivalent of college freshman courses. These programs are effective for students who have already decided that they

want to write software but do little to entice other students to explore software development career opportunities. The practical problems faced by most secondary schools attempting to implement these curricula include computing resource limitations and a lack of qualified teachers. These limitations further restrict the opportunities most students have to explore software development career opportunities.

With few exceptions such as the new Toronto curriculum, most secondary school programs fail to address the issues motivating the research described here, namely:

1. providing students with a broad understanding of software development early in their education
2. dispelling myths about software development careers
3. motivating students (especially females and minorities) to further study in this field

What are needed in most secondary school curricula are new curriculum modules providing students with the necessary breadth to understand software development career opportunities. These modules could form the basis of new courses or be integrated into existing information technology classes. The modules must be designed to be taken early in the student's program and complement existing advanced placement computer science courses if the high school is fortunate enough to possess resources to offer them. The modules must provide an exciting experience that captures the imagination and interest of the students in the same manner as strategy-based video games. One of the findings of a recent NSF funded study to increase participation of minorities in the computing disciplines supports this position with the recommendation to: "develop new more human-oriented and intellectually exciting high school computing curricula."[11] The new modules must be designed to be taught in less fortunate schools with limited resources where an offering may be the only opportunity students have to learn about software development.

III. Problem Statement

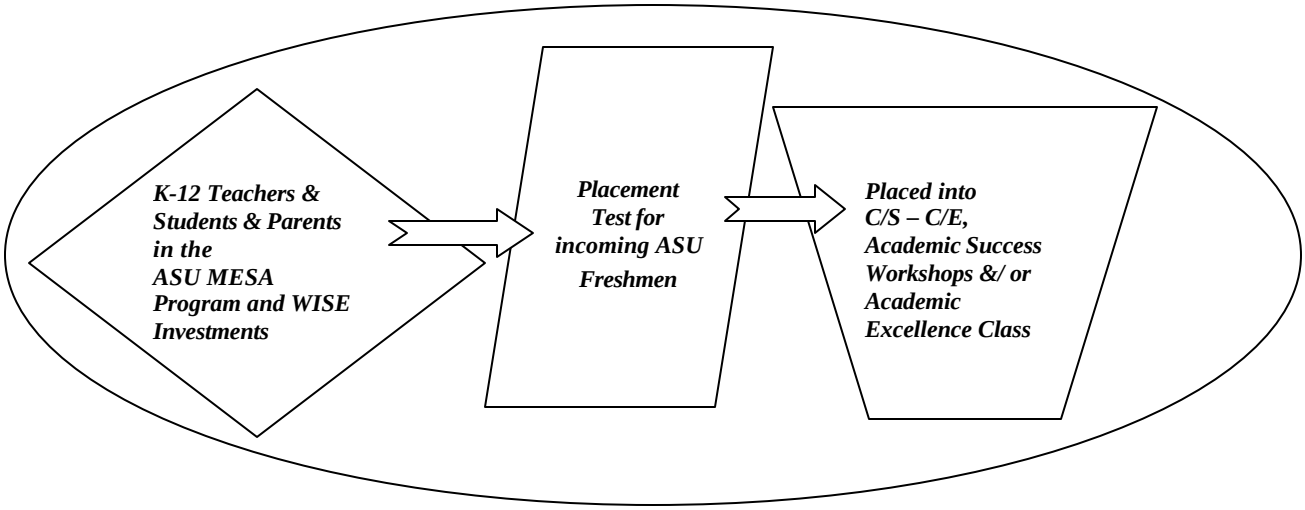
The goal of this project was to continue the development and refinement of an innovative secondary school software development program and of the curriculum modules developed in this pilot program. The specific goals are to provide students with a better understanding of the software development field, to dispel misconceptions, and to increase the number and diversity of students continuing their studies in preparation for software development careers. These goals are to be carried out through recruitment and retention programs.

At the same time, students, including many underrepresented minorities, come to the CEAS unprepared with the computer skills expected in ECE 100, CSE 100, and CSE 110. For example, many American Indian students from the reservations are not really prepared to embark on a computer related major given the fact, that they have not been exposed to the basics in computer applications, let alone major in computer science. Often the student realizes too late that they are lacking in computer skills and by the time they catch up, the semester is over and the course has to be repeated. These students need a means (placement exam) of determining for which of the beginning classes requiring computer skills they are prepared. Based on the placement exam,

some students need a basic computer class to get them up to speed to commence with their CEAS major. Additionally, there is a need for concept building and additional insight into the CSE 100 and CSE 110 classes. On average, per term, 36.5% of the students get a B or better in CSE 100 and 44.2% of the students get a B or better in CSE 110. For underrepresented minority students, however, only 28.2% received a B or better in CSE 100 and only 24.6% received a B or better in CSE 110. This gap is even wider when we consider that the underrepresented students grades are included in the CSE 110 classes' overall average. Academic Success Workshops will assist underrepresented students and others that need assistance with concept building in the CSE 110.

IV. Project Program Activities

COOL Project



The COOL Project is composed of several program activities. Here we will describe four of them and the progress made to date.

1. *COOL Teacher Institute*

The work described in this paper is a continuation of a piloting activity begun in the Summer of 2001. The pilot project was funded through a special legislative proposition in the state of Arizona (Proposition 301) designed to improve public education. The pilot project involved three Arizona high schools working with ASU personnel to develop and pilot software engineering curriculum materials at the high school level. The teachers in the program participated in a one-week software engineering workshop in which they learned software engineering concepts. The teachers then continued to work with ASU personnel in developing high school appropriate lesson plans for integrating the software engineering concepts into their current courses. The pilot project demonstrated the feasibility of teaching software engineering concepts at the high school level and the positive impact these materials had on the students. The pilot project also demonstrated the willingness of high school teachers to incorporate the materials into existing courses [8].

As a result of our pilot project experiences, a second year of funding was awarded to refine and expand the previous activities and create curriculum modules that can be disseminated on a much broader scale. During the Fall of 2002, 17 high school teachers were recruited to participate in the project. The participants were from a range of schools both public and private both large and small. The schools were chosen to represent different environments that the curriculum modules would need to be taught. During the Fall of 2002, a series of Saturday morning workshops were held with the team to instruct the teachers in software development processes. Throughout the workshop considerable time was spent in identifying the body of knowledge appropriate for secondary school students. An identification 21 high school appropriate software development lesson plans was also produced. The topics for each of the lesson plans is briefly described below:

1. Strategies for software debugging.
2. How to conduct a peer review or software inspection.
3. How to develop use cases and to document them in UML using Rational Rose.
4. Professional software computing societies.
5. How to work in a software development team.
6. Software maintenance in which students add new features to an existing software program. A variation on this is that the students switch programs and then perform the modifications.
7. Software component testing that involves testing techniques such as equivalence partitioning, boundary value, and statement coverage.
8. Project planning exercise involving work break down structures, pert chart and Gantt chart.
9. Software measurement exercise in which students keep track on how much time they spend on software development activities.
10. Software requirements lesson in which students learn how to work with a customer in eliciting and writing software requirements.
11. Software design exercise in object oriented software development.
12. UML lesson involving documenting a software design in terms of a class diagram and sequence diagrams using rational rose.
13. Organizing and participating in a technical meeting.
14. Rapid prototyping as a way of better understanding requirements.
15. Component based development in which students develop a program from existing components.
16. Incremental development.
17. Extreme programming, such as pair programming (two programmers write code on a single workstation together).
18. Engineering versus business computing careers (i.e., the difference between a computer science degree and a computer information systems degree).
19. How teachers perform assessment on team projects.
20. How to create a lessons-learned or post mortem for a project.
21. How to create component development folders.

2. The Placement Test for Incoming CEAS Freshmen

A few years ago, a mathematics placement exam was developed and put in place in cooperation with the Department of Mathematics. The purpose of this exam was to stem the high attrition rate in Mathematics, MAT 270 – Calculus with Analytic Geometry and also the low scores in that class. Our research showed that 75% of the students who receive lower than a C on their first mathematics class do not return for their second year.[12] Similarly, many students are having difficulty with the computer skills expected in ECE 100, CSE 100 and CSE 110. A similar computer skills placement exam was desired with diagnostics that would predict difficulty for the student in ECE 100 and additionally in CSE 100 and CSE 110. In summer 2002, we developed such a placement test and piloted it on the students in the Minority Summer Bridge Program and with students enrolled in one section of CSE 110 for the Fall 2002 semester. Students who did not score well in this exam were encouraged to enroll in a two hour Academic Excellence class on basic computer skills. The development and implementation of this test will allow CEAS advisors to appropriately place freshmen into computer science courses that they will be able to succeed in. This advising intervention and practice will address the high attrition rate and increase the retention rate for students majoring in the computer fields at ASU.

Entry computer science placement exams were examined and the following pilot placement exam is a compilation of original questions and questions from other placement exams.[13-21] See the following page for the pilot survey given. Students who scored 17-20 were deemed able to begin ECE 100 and CSE 110. Students who scored 13 or below were strongly advised to enroll in the Computer Basics course. Students who scored 14-16 were told that they would probably benefit from the Basics course.

To help determine which students in the Minority Summer Bridge Program should be advised to take the Computer Basics course, we asked the two student assistants, who worked with the students on their computer skills, including web design, to tell us who should take the course, who definitely did not need the course, and then to put the rest in a "maybe" category. In addition, we asked the students to assess whether they thought that they needed the course, weren't certain if they needed it, or if they knew that they did not need such a course. The Bridge students were then given the placement exam.

The evaluations were somewhat correlated. The student assistants agreed exactly on 11 of the 39 students and differed by more than one unit on only four. Of the twelve students who said they thought that they should take the Basics Course, seven were identified by at least one counselor as needing the course. Of the 13 Bridge students who chose to take the Basics Course, six ranked a 1 on the placement exam (needs the course), six ranked a 2 (maybe needs course), and one ranked a 3 (doesn't need course). Obviously, more work needs to be done to determine if the placement exam is a good predictor. Additional analysis will be done with the students' grade in the course and their subsequent success in taking other computer science classes.

ASU CEAS Pilot Computing Skills Placement Test

1. When it comes to computers, I characterize myself as:
   a. comfortable and confident
   b. good enough with things I know – nervous with new things
   c. largely clueless but willing to try
   d. hopeless – a total basket case

2. When sending an attachment with an e-mail message, the sender must
   a. Assign a device driver
   b. Provide a file
   c. Invert the distribution list
   d. None of the above.

3. Word processing software is used for:
   a. Document creation
   b. Insertion and deletion of text
   c. Modification of previous developed text
   d. All of the above.

4. A file saved in a word processor can never be changed for any reason whatsoever:
   a. True
   b. False

5. Incorrect entry of data on a spreadsheet will most likely be:
   a. Corrected by the program
   b. Corrected by the spreadsheet developer
   c. Corrected by the data entry individual
   d. Not possible to correct.

6. A browser is used to access:
   a. The Internet
   b. Random Access Memory (RAM)
   c. Cache memory
   d. All of the above.

7. URLs are used in computing for:
   a. Understanding a calculation
   b. Remembering program constants
   c. Locating a web page
   d. None of the above

8. A GUI is used for:
   a. Deterring computer users
   b. Entering data and receiving results
   c. Sticking keys on a keyboard
   d. All of the above

9. An operating system controls the use of computing resources:
   a. True
   b. False

10. An algorithm is a finite sequence of steps to solve a problem:
   a. Always
   b. Sometimes
   c. Never

11. The data that is stored on a disk:
   a. Must be refreshed hourly
   b. Can never be changed
   c. Remains unchanged until updated by a user
   d. None of the above

12. A computer system can be used to control the flight of an aircraft:
   a. True
   b. False

13. Which e-mail field the sender must always fill?
   a. Attachment field
   b. Subject field
   c. To field
   d. All of the above

14. The home page _____.
   a. Is the first page the user sees when starting a Web browser
   b. Is the Web page devoted to the advertisement
   c. Is set by the Web browser and cannot be changed
   d. All of the above

15. The Internet can best be described as a(n):
   a. network of computers
   b. network of computer networks all using the UNIX operating system
   c. network of computer networks
   d. network of computer networks all using the same operating system.

16. At acme.com there is a friend of yours named Jeff Smith. His UserId is Jsmith. What is most likely to be his e-mail address.
   a. smith@acme.com
   b. Smith@com
   c. Jsmith@acme
   d. Jsmith@acme.com

17. Web documents can contain:
   a. Graphical image
   b. Audio or sound
   c. Plain text
   d. All of the above

18. The software that gives the computer system the ability to display HTML documents is called _____.
   a. server
   b. client
   c. browser
   d. both A and B

19. Three control structures that are commonly used to develop an algorithm are:
   a. Sequence, selection, and iteration
   b. Start, pause, and stop
   c. Input, process, and output
   d. None of the above.

20. When you click a hypertext link,
   a. a pop-up box displays with a description of the hypertext link.
   b. the name of the linked file displays.
   c. another page or site on the Internet is accessed.
   d. you are automatically logged off the Internet.

3. The Academic Excellence Class (AEC)

The AEC addressed many aspects of getting the students computer literate and providing them with a vision for the time until they complete their degree program. Software tools that most high school students' use were covered early on during the course, such as email, word

processing, spreadsheets, and web searching. In addition, software development basics were covered in order to prepare the students for the first computer science course. For meeting the vision part of the course, Parallax BOE-BOT autonomous robots that are relatively easy to build were used for group projects in which the students modified the onboard program for robot motion. The Silicon Summit III identified the lack of access to Internet appliances as a major detriment to the success of underrepresented minority students [22]. This course placed wireless handheld Internet devices, Toshiba e740 Pocket PCs, in the hands of the students with the express purpose of measuring the effectiveness of this medium on student performance.

The major difference between this offering of the AEC and the prototype course from a year earlier was that more time was spent on software development issues and applications. This year the students needed the additional time on software development in order to develop BASIC programs for the BOE-BOTs. There was also a group project in which each three-member group formulated the software requirements and then designed and implemented a robot application. The group projects involved navigating a planned course that each group defined.

In addition, the Toshiba Pocket PCs allowed for the students to organize schedules, tasks, and notes, as well as access the Internet through the wireless LAN capability. This handheld device further exploited the applications that were previously accessed only on a PC.

Intermediate evaluations of the Computer Basics course were held at approximately one-month intervals. Short evaluation sheets asked the student to identify the most important topic of the week, the "muddiest" topic, and topics for which the students wanted more information. Students were also invited to give comments. This class, for the most part, understood the most important topic and few had "muddiest" topic comments. However, the additional information requested spawned areas of further study in careers, applications, and algorithms.

4. Academic Success Workshops (ASW)

An Academic Success Workshop started in Fall 2002 for students in CSE110 course, Principles of Programming with Java. The workshop was scheduled twice a week every Monday and Wednesday. By analyzing the difficulties that students were facing in the CSE110 course, the workshop provided different strategies and materials that would help students to learn the subject matter of this course.

The workshop focused on concept building and practice exams to assist students in the CSE110 course, which has been designated as a gatekeeper course.

The attendance to the workshop was quite encouraging. The average of the number of students attending the workshop was 12, with this number rising up to 32 at times. Students were asked to work in small groups. Students in the workshop were free to ask questions and they took full

advantage of this informal environment. The focus was in finding the problems that individual students were facing and helping them to overcome the problems. The strategy was on strengthening the basic skills, such as teaching objects, classes, and problem solving. Selecting interesting and challenging problems, and encouraging them to discuss it between themselves accomplished this goal.

At the end of the workshop we provided students with a survey, and sixteen students completed it. The result of the survey has been extremely positive. The response to the question asking if the materials covered in the workshop helped them to understand the concepts well, 75% responded that it was very helpful. When asked if the exam reviews were helpful, 92% responded positive. Most importantly, 100% of the students strongly recommended continuing this workshop for the following semesters. The survey revealed that the workshop strategy is a success, and it has a significant impact on student achievement.

V. Discussion and Future Plans

During the Spring of 2003 the lesson plans will be developed by the COOL Teacher Institute and taught by the high school team. To assist in the development of the materials and their instruction, undergraduate students will be brought into the project. Each will have already been through at least one software engineering course. One undergraduate student will be assigned to each of the teachers to assist them in the development and instruction of the curriculum materials.

Mentoring of the teachers will continue through the project and culminate with another workshop at the end of the project to assess the lessons learned. All materials will be gathered and organized on a project web site.

We plan to work closely with the secondary school teachers, their school curriculum committees and ASU educators in CRESMET (Center for Research in Education in Science, Mathematics, Engineering and Technology) to develop and utilize detailed assessment procedures. These procedures will include pre and post tests of student knowledge and perceptions of the software development field as well as statistics on the impact of the curriculum modules on students further pursing their studies in this area. A special emphasis will be placed on assessing the impact of the curriculum modules on the diversity of the student population.

The Placement Exam for Computer Science will be furthered studied and modified. Students in one section of the Fall 2002 CSE 110 class were given the placement exam at the beginning of the semester. Two students were advised to take Computer Basics based on their score. The placement scores of all of these students will be correlated with their final grade in their computer science class and their participation in the Academic Success Classes and reported in a future paper. The placement exam will be given again this Spring 2003 semester in both sections of CSE 110 and the scores correlated with the grade in the class and participation in the Academic Success Classes.

The Academic Excellence class was well received in Fall 2002. It addressed one of the main complaints of the one-hour pilot course by being a two-hour credit class. The students were excited to have the use of Pocket PCs. We are looking for funding to support the students

retaining the Pocket PCs for their personal use and then doing a longitudinal study to determine what effect, if any, these devices had on their retention in engineering. We plan to offer the class again in Spring 2003.

The Academic Success Classes were rated very highly by the students that attended them. Changes to be made in Academic Success Classes for Spring include: encouraging more students to participate in the workshop, finding strategies to make students want to attend all sessions of the workshop, and making the classes more enjoyable to participants by getting them more involved in group learning activities.

Although the high school and university level activities seem very loosely connected at this point, the COOL team has found advantages in working together. As we refine the computer placement exam, we will gain useful information to pass on to high school teachers of computer science so they can understand what is expected of entry computer science students. Also, the COOL team has been able to give curriculum advice to industry volunteers who will present a summer program for disadvantaged middle school students on computers and skills needed for the successful navigation of high school and college. This program is one of several sponsored by our college to help increase the enrollment and diversity of our engineering and computer science students. Through these programs we are administering the computer placement exam and giving heads up counseling on the possible need of the students for a basic course in computers before taking a Java course.

The project has been funded through Proposition 301 funds in the state of Arizona.

References

1. J. Margolis & A. Fisher, *Unlocking the Clubhouse – Women in Computing*, Cambridge Massachusetts, The MIT Press, 2002.
2. S. Merritt, et.al., "ACM Model High School Computer Science Curriculum", http://www.acm.org/education/hscur/index.
3. J. Urban, M. Reyes, and M. R. Anderson-Rowland, " Minority Engineering Program Computer Basics with a Vision," *ASEE/IEEE Frontiers in Education Conference*, Boston, MA, November 2002, CD-ROM, 5 pages.
4. M. Reyes, M. Anderson-Rowland, & M. McCartney, "Learning from our Minority Engineering Students: Improving Retention," *American Society for Engineering Education Annual Conference*, St. Louis, Missouri, June 2000, CD-ROM, 8 pages.
5. C. Jordan & M. Anderson-Rowland, "Evaluating the Success of an Underrepresented Minority Summer Recruitment Program for Rising Juniors and Seniors," *Joint NAMEPA/WEPAN National Conference*, Alexandria, Virginia, April 2001, pp. 17-21.

6. D. Newell, W. Fletcher, & M. Anderson-Rowland, "The Women in Applied Science and Engineering Program: How Diversified Programming Increases Participation," *American Society for Engineering Education Annual Conference*, Montreal, Quebec, Canada, June 2002, CD-ROM, 9 pages.

7. D. Newell, S. Fletcher, & M. Anderson-Rowland, " The Women in Applied Science and Engineering (WISE) Recruitment Programs: Investing in the Future," *ASEE/IEEE Frontiers in Education Conference*, Boston, MA. November 2002, CD-ROM, 5 pages.

8. J. Collofello, "Creation, Deployment and Evaluation of an Innovative Secondary School Software Development Curriculum Module," *ASEE/IEEE Frontiers in Education Conference*, Boston, MA. November 2002, CD-ROM, 4 pages.

9. J. Lan, "An Analysis of the Evolution of Computer Education in Shanghai Secondary Schools: Relevant to American Schools?" http://www.coe.uh.edu/insite/elec_pub/HTML1997/pt_lan.htm

10. "Toronto Computer Science Curriculum, http://webschool.dpcdsb.org/acse/resources/kitto/cmpsci0.htm

11. B. York, et.al., "Final Report of the Workshop on Increasing Participation of Minorities in the Computing Disciplines", http://www.ccs.neu.edu/home/york/aascc/nsfmi_prop/nsfmi_rep.html

12. M. Anderson-Rowland, "The Effect of Course Sequence on the Retention of Freshmen Engineering Students: When Should the Intro Engineering Course Be Offered?" *ASEE/IEEE Frontiers in Education Conference*, Tempe, Arizona, November 1998, pp. 252-257.

13. University of Maryland University College Center for Teaching and Learning, *Basic Computer Skills Tutorial for a Windows Environment*, http://www.umuc.edu/distance/odell/ctla/basic_skills/basic_skills.html, 2002.

14. Miami University, Incoming Students Basic Computer Skills and Self-Assessment, http://www.units.muohio.edu/mcs/suppctr/lis/mcistraining/proficiencies.shtml , 2002.

15. Teaching Universal Skills, Basic Computer Skills Quiz, http://www.learningtrain.net/Basic%20Computer%20Quiz.htm, 2001.

16. University of California. Santa Barbara Computer Engineering Program, Self-Placement Test for Computer Science Courses at UCSB, http://www.ce.ucsb.edu/Self-Placement%20Test.pdf, 2002.

17. University College of the Fraser Valley Student Services, Assessment Services – Computer Skills Placement Test – CSPT, http://www.ucfv.bc.ca/stuserv/cas/CSPT%20info.htm, 2002.

18. North Carolina Department of Public Instruction, Tests of Computer Skills , http://www.ncpublicschools.org/testing/, 2002.

19. University of Wisconsin Colleges, UW Colleges Computer Skills Test - Word Processing, http://www.uwc.edu/DistEd/DETest/, 2002.

20. University of Houston – Clear Lake Instructional Technology, Basic Computer Literacy Skills, http://inst.cl.uh.edu/GettingStarted/CompTest.htm, 2002.

21. University of North Texas, Student Orientation to Distributed Learning – Self-Test #2: Computer Literacy, http://courses.unt.edu/orientation/web_home/Basic_Web/web_A.1.3.2.html, 2002.

22. MSNBC News, Technology - Silicon Summit III, http://www.msnbc.com/news/siliconsummit_front.asp, 2002.

Biographies

JAMES S. COLLOFELLO
James S. Collofello is Professor and Associate Chair of the Department of Computer Science and Engineering at ASU. He earned his Ph.D. from Northwestern University. His research and teaching interests lie in the software engineering area with an emphasis in software project management and software quality assurance. He contributes regularly to the computer science education literature and has been active in high school outreach programs.

JOSEPH E. URBAN
Joseph E. Urban is a professor of computer science and currently serves the CEAS Inclusive Learning Communities program director at Arizona State University. His research areas include software engineering, computer languages, data engineering, and distributed computing. He has received the IEEE Computer Society's Meritorious and Distinguished Service Awards, a Distinguished Professor Award while at the University of Louisiana at Lafayette, and an Association for Computing Machinery Doctoral Forum Award.

MARY R. ANDERSON-ROWLAND
Mary R. Anderson-Rowland is the Associate Dean of Student Affairs in the CEAS at ASU. She earned her Ph.D. from the University of Iowa. Her awards include the YWCA Tribute to Women 2001 Award (Scientist/Researcher) and the University Achievement in Gender Equity Progress Award, Faculty Women's Association, 1995. She was named an ASEE Fellow in 2001, one of "30 Prominent Women in Phoenix" Award in 2002, and the Society of Women Engineer's Distinguished Engineering Educator in 2002.

FAYE NAVABI received her B.S. and M.S. in Computer Science from University of Louisiana at Lafayette. She has been a full time lecturer at the department of Engineering and Computer Science at Arizona State University since 1997. She also taught at the University of Louisiana at Monroe for four years. During her teaching career she has taught a wide variety of courses including: Object Oriented Programming using Java, Data Structures and Structured Programming, Internet, Principle of Programming using Visual Basic, and Computer Literacy.

DORIS ROMAN is the Director of Student Outreach and Retention Programs for the CEAS. Prior to ASU, she held positions at San Francisco State University, University of Notre Dame, Cornell University, and Rensselaer Polytechnic Institute. She has extensive experience in the administration of comprehensive support services for students. Doris works with AISES, ASPIRA, NSBE, MAES, SACNAS, SHPE, STEP, and SWE. She also works with government and industry developing programs and addressing the educational pipeline for students in STEM.